REAL USERS, REAL DEVICES, REAL TIME
Powered by Crowd & Cloud Technologies

A complete Selenium Techstack – From conception to evaluation with open source software
Test Automation at a glance

Test Design

Test Development

Test Execution

Test Evaluation
Test Design

The Challenges

1. Legibility
   Accelerates error analysis, ensures broad common understanding and thus promotes cooperation.

2. Structure
   Promotes logical structure, for example by precondition, action and assurance.

3. Reusability
   Reduces potential sources of error, improves overall maintainability and increases scalability.

4. Abstraction from the test object
   Using declarative instead of imperative test case descriptions shifts focus to functional rather than technical aspects.
Test Design

Tools in our stack

- **Legibility**
  - BDD with Cucumber
  - Human-readable test specification thanks to Gherkin Syntax
  - Involves the entire team, from tester to developer to PO

- **Reusability**
  - Step Definitions by Gherkin Syntax
  - Flexibly combinable

- **Structure**
  - Logical structure according to Given - When - Then thanks to Gherkin template
  - Supplementary explanations with the help of narratives

- **Abstraction from the test object**
  - Gherkin as a natural additional level of abstraction
Test Automation

Test Automation at a glance

Test Design
- Legibility
- Structure
- Reusability
- Abstraction from the test object
  Gherkin Syntax as an interface

Test Development

Test Execution

Test Evaluation
Test Development

The Challenges

Stability
Reduces the number of so-called false negatives, speeds up individual test runs.

Best Practices
Prevents the use of bad practices (such as static waits), promotes good practices such as the Page Object Pattern.

Reusability
Reduces potential sources of error, improves overall maintainability and increases scalability.
Test Development

Tools in our stack

- **Reusability**
  - Selenium Browser Testing with Selenide
  - Simplest test configuration
  - User-friendly session and driver management
  - Always updated framework methods

- **Stability**
  - Cross-browser proven framework methods
  - Possibility to communicate directly with the backend

- **Best Practices**
  - Implicit, globally configurable timeouts before each operation
  - Native page object integration
  - Integrated error handling, screenshots and DOM dumps
Test Automation

Test Automation at a glance

Test Design
- Legibility
- Structure
- Reusability
- Abstraction from the test object
  Gherkin Syntax as an interface

Test Development
- Simple code with clear structure
- Stable Selenium framework
- Best Practices
  Selenide Framework, Best Practices

Test Execution

Test Evaluation
Test Execution

The Challenges

Predictability
Event and time based execution of E2E tests

Parameterizability
Execution of the same test suite in different test environments and in different browser operating system combinations.

Connecting data
Execution of the E2E tests only after successful build (including execution of the unit tests) and deployment on the test environment.
Test Execution

Tools in our stack

- **Execution of the tests in an independent system (Continuous Integration, CI)**
  - Jenkins with Jenkins Pipelines
  - Parameterization of the pipelines in Jenkinsfile incl. schedule possible
  - Own projects as required for special test runs (e.g. nightly-main-browsers)

- **Loose coupling of product and test development**
  - Jenkinsfile in every git on every branch
  - Parameterization allows easy coupling, e.g. the URL (staging-123.tests.local)

- **Backgrounds and benefits of this tool choice**
  - Test development of unit tests is part of every feature
  - E2E tests can "hang behind" for time / cost reasons or achieve less coverage
  - Jenkinsfiles in Git define the complete build - (almost) no configuration in Jenkins
## Test Automation at a glance

### Test Design
- Legibility
- Structure
- Reusability
- Abstraction from the test object
  - Gherkin Syntax as an interface

### Test Development
- Simple code with clear structure
- Stable Selenium framework
- Best Practices
  - Selenide Framework, Best Practices

### Test Execution
- Smoke tests after each build
- Time-controlled long test runs
- Parameterization of builds
- Different test environments
  - Jenkins Pipeline

### Test Evaluation
Test Evaluation

The Challenges

Assessability
Freely determinable evaluation and error category of test run errors, automatic assignment for frequently recurring errors.

Structurability
Sensible grouping of test results by product version, test environment and browser operating system combination.

Visualizability
Graphical processing of test results, configurable dashboards, automatic error notifications.

Analysability
Integration of stack traces, error logs, screenshots and video recordings, link to specification and issue trackers.
Example: 200 test cases from 10 feature groups of a web portal (horizontal in the matrix)

Test runs on all common desktop browsers (vertical)
- Win 10: IE11, Edge, Chrome, Firefox, Firefox ESR
- macOS: Safari, Chrome, Firefox, Firefox ESR
- Win 7: IE11 (strong OS dependency)
- = 10 Test environments/Browser

→ 200x10 matrix with 2,000 test results per test environment

5 feature branches + Staging = 6 branches / environments to be tested
= 12,000 test results per night
Test Evaluation

Tools in our stack

- **Many tests, a lot of data → Jenkins plugins or similar often inadequate**
  - Own tool for the evaluation of all test runs and statistical correlation
  - Delivery of additional data to the tool (not just junit-results.xml)
  - Integration of test or system specific data by the test code, e.g. Resolve UUID error in backend and extract stacktrace

- **Statistically, errors occur even with very good stability of the system**
  - Quote of a developer: "The system has to recognize outliers easily!“
  - Solution: "AI" or automatic data analysis with machine learning
  - Tags, feedback from users and more data feed the system for future analysis
  - Checks of system availability

- **reportportal.io provides many of the desired functions here**
  - Automated analysis and easy integration into the test suite
  - Dashboards and clear UI
## Test Automation

### Test Automation at a glance

<table>
<thead>
<tr>
<th>Test Design</th>
<th>Test Development</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Legibility</td>
<td>- Simple code with clear structure</td>
</tr>
<tr>
<td>- Structure</td>
<td>- Stable Selenium framework</td>
</tr>
<tr>
<td>- Reusability</td>
<td>- Best Practices</td>
</tr>
<tr>
<td>- Abstraction from the test object</td>
<td></td>
</tr>
<tr>
<td>Gherkin Syntax as an interface</td>
<td>Selenide Framework, Best Practices</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test Execution</th>
<th>Test Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Smoke tests after each build</td>
<td>- Correlation with past tests</td>
</tr>
<tr>
<td>- Time-controlled long test runs</td>
<td>- Extension of the reports</td>
</tr>
<tr>
<td>- Parameterization of builds</td>
<td>- Screenshots, Backend information</td>
</tr>
<tr>
<td>- Different test environments</td>
<td>- Automated analysis</td>
</tr>
<tr>
<td>Jenkins Pipeline</td>
<td>reportportal.io with connection in the code</td>
</tr>
</tbody>
</table>
GitHub: https://github.com/Testbirds-GmbH/selenium-techstack
Felix Kuperjans | Technical Lead Device Cloud
Testbirds GmbH | Radlkoferstraße 2 | 81373 Munich
+49 89 856 33 35 – 0
f.kuperjans@testbirds.de